I J C T A, 9(17) 2016, pp. 8729-8738

Net App Driver using Copr HD SouthBound SDK

Kshitiz Saxena*, Richa Gupta* and Karanjeet Singh Arora*

ABSTRACT

An unprecedented data growth is being observed around the world today. Data needs are growing at a rate unsustainable with today's infrastructure and labor costs. This growth presents new opportunities as well as a host of new challenges-cost effective controlling, managing and accessing the data. Heterogeneity of storage devices add on to these challenges. To keep pace with this explosion in data and to cope up with the challenges, the current storage infrastructure needs to be massively scalable, locally and remotely accessible, provide continuous uptime and should have the efficiencies of automation. Software-Defined-Storage (SDS) comes as a rescuer which brings the benefits of 'cloud' to storage and delivers a scalable, cost effective solution to meet the needs of tomorrow's data centres. One such platform is CoprHD, pronounced as 'Copper Head', which gives storage administrators a chance to benefit from Cloud-Based Storage capabilities. For improvements in this self-service automation software, an open-source community is working towards the addition of third party arrays to CoprHD. For writing drivers for these third party storage systems, CoprHD provides storage driver Southbound SDK in which various functionalities have to be implemented. Once the third party arrays have been configured to meet CoprHD's requirements, CoprHD system administrator defines the virtual pools on them. The tenant admin who creates the storage resources for users use these virtual pools for providing various storage services. This way it becomes possible to have storage infrastructure comprised of different array types, often from different vendors so as to deliver a wide variety of storage services.

Keywords: cloud computing, storage cloud, virtualization, CoprHD Storage Controller

I. INTRODUCTION

CoprHD is used for building a storage cloud, a contribution of EMC[2] in the open source market. CoprHD aims to discover pools and automate the management of heterogeneous storage systems. It itself does not provide storage instead it holds the inventory of the storage systems in data center and understands their connectivity. It classifies the storage using policies and provides end-to-end storage automation via a self-service catalog.

CoprHD is designed with two key goals:

1. Make an enterprise or a service provider datacenter full of storage resources look and feel like a single massive virtual storage array, automating and hiding its internal structure and complexity.
2. Extend a full set of provisioning operations to end-users ranging from simple operations such a block volume creation, to fairly complex operations such as creating disaster-recovery-protected volumes in different data centers in a truly multi-user, multi-tenant fashion by providing tenant separation, access control, auditing, usage monitoring, and other features[1].

Since CoprHD is released as an open-source product, it is expected to attract attention from 3-rd party storage vendors and it is anticipated that vendors will start integrating their arrays into CoprHD. This

* Department of Computer Science & Engineering, Bharat Institute of Technology, Meerut, UP, India, *E-mail: hodcse@bitmeerut.co.in; richagupta2107@gmail.com; karansingh392@gmail.com*

requires to provide clear way to integrate support for storage array drivers into CoprHD. Our work aims to add support specifically for NetApp storage drivers using CoprHD Southbound SDK.

II. LITERATURE REVIEW

a) Cloud: Cloud computing refers to the provision of computational resources on demand via a computer network. In this model, the user's computer may contain almost no software or data (perhaps a minimal operating system and web browser only) whereas in the traditional model of computing, both data and software are fully contained on the user's computer. "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (example- network, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (NIST)[2]. The IT challenges listed below have made organizations think about the Cloud Computing model to provide better service to their customers:

- Globalization: IT must meet the business needs to serve customers world-wide, round the clock.
- Aging Data Centers: Migration, upgrading technology to replace old technology.
- Storage Growth: Explosion of storage consumption and usage.
- Application Explosion: New applications need to be deployed and their usage may scale rapidly. The current data center infrastructures are not planned to accommodate for such rapid growth.
- Cost of ownership: Due to increasing business demand, the cost of buying new equipments, power, cooling, support, licenses, etc., increases the Total Cost of Ownership (TCO).
- Acquisitions: When companies are acquired, the IT infrastructures of the acquired company and the acquiring company are often different. These differences in the IT infrastructures demand significant effort to make them interoperable.

Cloud Computing has become a buzz word due to the growing needs to build complex IT infrastructures. Outsourcing computing platforms is a smart solution for users to handle complex IT infrastructures.

(b) Software Defined Storage: The movement to smarten up every aspect of IT infrastructure rolls ever onward, Software-defined storage (SDS) is the next big trend.[3] It represents an important and a promising next step towards helping organizations address the myriad storage and data challenges they face today as these challenges are inadequately being addressed by traditional storage. The cliché that most IT organizations have been experiencing tremendous growth in storage-related demands for capacity, performance, functionality, and flexibility is not only true, but it is also reaching a breaking point with traditional storage approaches. SDS is an approach to data storage in which the programming that controls the storage related tasks is decoupled from the physical storage hardware. It places emphasis on storage-related services rather than storage hardware.

A key enabler of the SDS architecture is an SDS controller. It has a number of characteristics:

- decrease the provisioning time
- reduce the complexity of managing data
- reduce the cost of storage operations and management
- architecturally scalable and extensible
- provide greater visibility and transparency into the managed storage systems

(c) Software Defined Network

Software-defined networking (SDN) is an approach to computer networking that allows network administrators to manage network services through abstraction of higher-level functionality.[4] This is done

by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination.

SDN is a compelling development for the public sector. It helps to simplify operations by automating and centralizing network management tasks. It makes the network more responsive to dynamic business and institutional needs by coupling applications with network control.

One of the primary advantages of software defined networking is that it creates a framework to support more data-intensive applications like big data and virtualization. Some of the specific advantages of software defined networking:[5]

1. Centralized network provisioning.
2. Holistic enterprise management
3. More granular security
4. Lower operating costs.
5. Hardware savings and reduced capital expenditures.
6. Cloud abstraction.
7. Guaranteed content delivery.

SDN is an architecture purporting to be dynamic, manageable, cost-effective, and adaptable, seeking to be suitable for the high-bandwidth, dynamic nature of today's applications. SDN architectures decouple network control and forwarding functions, enabling network control to become directly programmable and the underlying infrastructure to be abstracted from applications and network services.

The SDN architecture is:

- Directly programmable: Network control is directly programmable because it is decoupled from forwarding functions.
- Agile: Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.
- Centrally managed: Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.
- Programmatically configured: SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs, which they can write themselves because the programs do not depend on proprietary software.
- Open standards-based and vendor-neutral: When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.

(d) Virtualization

Virtualization means to create a virtual version of a device or of a resource, such as a server, storage device, network or even an operating system where the framework divides the resource into one or more execution environments[6]. Even something as simple as partitioning a hard drive is considered virtualization because one single drive is partitioned to create two separate hard drives. Devices, applications and human users are able to interact with the virtual resource as if it were a real single logical resource. The term virtualization is now associated with a number of computing technologies including the following:

- Storage virtualization: The amalgamation of multiple network storage devices into what appears to be a single storage unit.
- Server virtualization: The partitioning a physical server into smaller virtual servers.
- Operating system-level virtualization: A type of server virtualization technology which works at the operating system (kernel) layer.
- Network virtualization: Using network resources through a logical segmentation of a single physical network.

Additionally, virtualization can also be of the following types-data storage virtualization for combining local & network resources, storage virtualization for grouping physical storage devices into a single unit, improving availability using virtualization for reaching high level of availability, improving performance using virtualization using stripping and caching and also capacity improvement [7].

Virtualization technology makes it easy to manage the resources. It abstracts and isolates the underlying hardware and networking resources in a single hosting environment. It increases the security of cloud computing by protecting both the integrity on guest virtual machine and cloud components virtualized machines which can be scaled up or down on demand and can provide reliability. It provides resource sharing, high utilization of pooled resources, rapid provisioning and workload isolation.

There are three key steps involved in making resources available to consumers. They are:

1) Deploying virtualization software

2) Discovering resource pools

3) Creating virtual resources

The virtualization software performs the abstraction of the physical resources and are deployed on compute systems, network devices and storage devices. The key functions of a virtualization software is to create resource pools and create virtual resources. A resource pool is an aggregation of computing resources, such as processing power, memory, storage, and network bandwidth, which provides an aggregated view of these resources to the control layer.

For example, storage virtualization software pools capacity of multiple storage devices to appear as a single large storage capacity. Similarly, by using compute virtualization software, the processing power and memory capacity of the pooled physical compute system can be viewed as an aggregation of the power of all processors and all memory.

There are many benefits of virtualization like:

- Easier manageability
- Elimination of the compatibility issues
- Fault isolation
- Increased security
- Efficient use of resources
- Portability
- Problem free testing
- Rapid deployment
- Reduce costs, etc.

The recent trends in virtualization are consolidation of data centers thus reducing the managing cost. Apart of its benefits it has some drawbacks like managing virtual resources is critical and migrating services of these resources are difficult in achieving high availability.

On a server failure, VM will be restarted on the other virtualized server in resource pool restoring the required services with minimum service interruption.

Virtual resources are critical for managing and data monitoring. Running applications with high utilization and availability is a challenging issue.

(e) ViPR Controller

The relationship between the consumers and providers of IT has been turned on its head. With the advent of public cloud 'as-a-service' offering, circumventing IT no longer requires an engineering degree, a high-end Windows or Linux server and expensive software. All it takes is an Internet connection and a credit card. The ease with which developers and end users can find readily available technology alternatives means they can hold enterprise IT departments and traditional IT service providers to a much higher standard for service and delivery. Fair or not, IT departments and service providers find it increasingly difficult to dictate technology choices to their internal and external customers; they have to compete for it[8].

ViPR Controller is a storage automation software based on the open source development project CoprHD. It is a lightweight, software-only solution that transforms the existing storage environment into a simple and extensible platform that can deliver fully automated storage services[9]. ViPR Controller abstracts multi-vendor storage from physical block and file arrays into a pool of virtual shared storage resources as well as abstracts the storage control path from the underlying hardware arrays so that access and management of multi-vendor storage infrastructures can be centrally executed in software. It makes a multi-vendor, heterogeneous storage environment look like one, big shared storage platform. It uses software adapters that connect to the underlying arrays. This is similar to how device drivers enable compatibility of any device with a personal computer- a 'plug and play' storage environment. ViPR also have a suite of data services such as snapshot, replication, data migration, movement etc. The controller provides the ability to manage storage infrastructure (control plane) and the data residing within that infrastructure (data plane).

ViPR Controller features block and file control services that provide all the functionality of physical block and file storage arrays as virtual services. Block and file control services allow users to manage block volumes, NFS file systems or CIFS shares, and advanced protection services such as snapshots, cloning, replication and high availability. These block and file control services offer full storage functionality as if the user were accessing a physical array. In contrast, block storage volumes typically provided for use by virtual compute instances in public clouds can forfeit many advanced array features in favor of using commodity disks for lower cost and operational simplicity. ViPR Controller, however, does not require that sacrifice, delivering operational simplicity and maintaining all the advanced features of the arrays such as mirrors, clones, snapshots, and multi-site high-availability, and replication.

ViPR Controller guides an administrator through the process of discovering the storage infrastructure. Once a storage administrator adds arrays, they are automatically discovered, including all their corresponding storage pools and ports. Once the Fibre Channel switches are added, they are automatically discovered and mapped to the Fibre Channel networks.

ViPR Controller hides the complexity of all the underlying storage arrays and exposes their core functionality as services while retaining the unique attributes of the arrays. Storage administrators then create Virtual Storage Pools in ViPR Controller that represent sets of capabilities required by unique application workloads. ViPR Controller simplifies and automates repetitive storage provisioning, data protection and management tasks by virtue of an abstracted, central control path. This extensible platform enables an organization to develop new services and adapters to support additional arrays[10].

(f) Southbound SDK

Southbound Software Development Kit (SDK) is specifically designed to make it easier for storage vendors and other third parties to add support for other storage systems to CoprHD. A southbound interface is a component's lower level interface layer. It serves as an alternative protocol specification in Software Defined Networking (SDN). The CoprHD acts as a 'storage manager' and perform actions like discovery, provisioning, metering and monitoring. The term is Southbound because the target storage system is in the south of the CoprHD. ViPR sits in between providing storage services and management to the northbound servers leveraging the Southbound storage.

The ultimate goal of the Southbound is to provide an easy plug ability of device drivers into CoprHD. It will ensure that there is no dependency on the CoprHD infrastructure service and also it will support integrating device specific orchestrations into CoprHD.

(g) NetApp Filer

NetApp filer is also termed as NetApp Fabric Attached Storage (FAS) or NetApp's Network Attached Storage (NAS) device. These are the devices that are an offering in NetApp's area of storage systems. NetApp FAS is a type of disk storage device which owns and controls the file systems, present files and directories of the network using Data ONTAP as an operating system.

III. ISSUES FACED IN INTEGRATING THIRD PARTY DEVICE DRIVERS

CoprHD greatly simplifies discovery and self-service provisioning of storage services. However, it has certain limitations.The outsourcing of CoprHD paved way for a number of third party vendors to integrate their storage arrays in it. In the past, integration of new type of storage array to CoprHD was done internally by CoprHD engineering. Typically, new array types bring their own flavor of data services, unique capabilities, their own device export constructs. All these were primarily statically coded in CoprHD. This required additions and modifications to several software components within CoprHD core — UI, persistent classes, export orchestrators, placement matchers. There was no focus on clean separation of CoprHD core from device driver layer. Practically, layer of CoprHD which talked to device APIs also performed CoprHD database updates, managed CoprHD task completion and executed other internal task communicating directly to CoprHD core. The focus was on utilizing device APIs to provide rich set of functionality to clients. Isolation of device access to its own layer and avoiding static coding device dependency into product core was not a goal at that time.

Following points confront the present limitations of current device access layer in CorpHD.

- Interface Definition: Declaration of device layer interfaces uses CoprHD persistent classes, primary keys of persistent classes and requires task completer instance with specific implementation to handle completion of the request. This implies that the device layer has direct access to CoprHD database. Typical task completer in the call to device layer should update CoprHD database with physical information about provisioned resource and notify core about request completion. When operation is part of multistep workflow, task completers call workflow completer, which in its turn sends request to core workflow service to complete workflow step and to initiate other workflow steps which have been blocked on the completed step. Task completers are operation specific. From this description it is clear that declarations of device access layer in CoprHD have direct dependency on CoprHD infrastructure.
- Interface Implementation: Implementation of device layer API is tightly bound to Cassandra and Zookeeper, and in some cases needs access to instances of core CoprHD services for internal communication. Typically, implementation of these interfaces uses parameters, which reference

persistent classes, to read necessary data from CoprHD database, builds request to physical device, executes this request and calls task completer to update database with physical properties of provisioned devices and to call CoprHD core services to notify about task completion.

The major issue with this approach is that interfaces and their implementation require knowledge of internal details about CoprHD persistent layer and CoprHD core services.

- Device Discovery: Currently CoprHD does not have generic API and generic model to discover and to process data services and capabilities provided by storage devices in device independent way. Most of the capability information is statically coded in the persistent classes and across internal CoprHD services and CoprHD UI. The issue here is that unique capabilities of new storage arrays should be manually coded into CoprHD database, core services and UI.

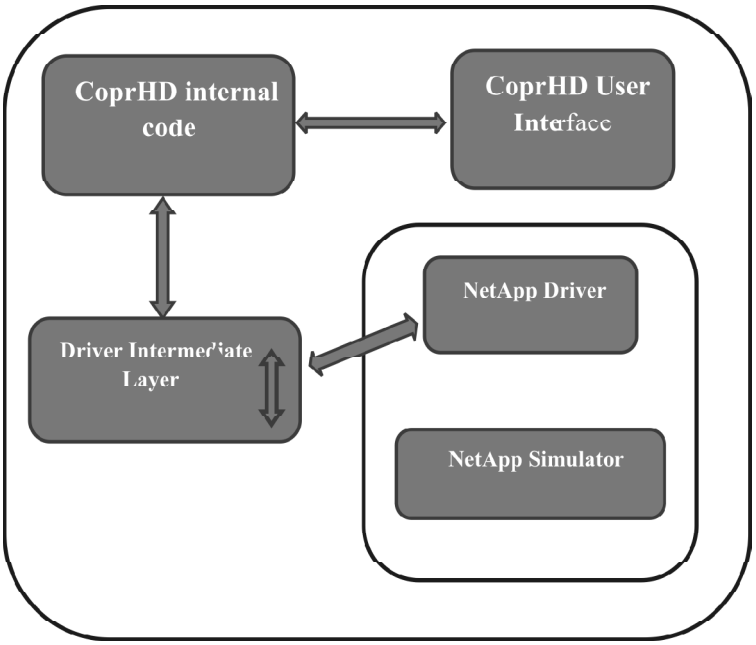

Figure 1: High-level implementation details

IV. EXPERIMENTAL EVALUATION

1. Functional Requirements

1.1 CoprHD's User Interface-

This is the first step prior to adding support for NetApp storage arrays. The user interface provides a convenient way to access numerous functionalities of the SDS controller.

1.2 NetApp Simulator-

It is not always possible to have the physical hardware present, nor is it convenient to use it for a variety of purposes, for example, testing. Therefore, for such purpose, we have simulators that give us the same look and feel and allow us to test functionality and perform tests. Here, since we are trying to add support for NetApp storage, we use a NetApp simulator.

1.3 REST Client-

REST stands for Representational State Transfer. It relies on a stateless, client-server, cacheable communications protocol — and in virtually all cases, the HTTP protocol is used. To get the response of several APIs performing a number of operations, we use a REST client which is added as a plugin in our browser.

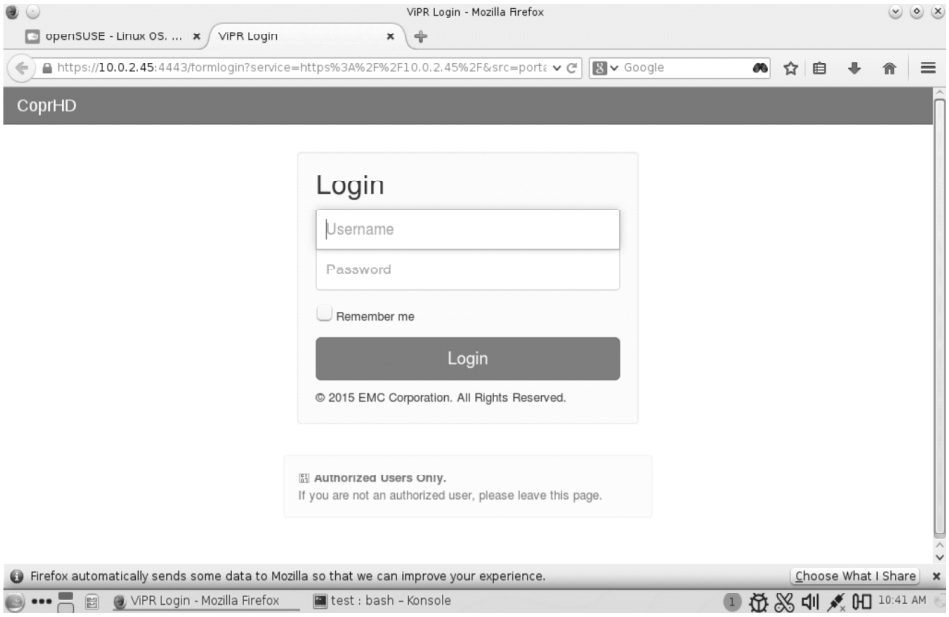

Figure 2: CoprHD's User Interface

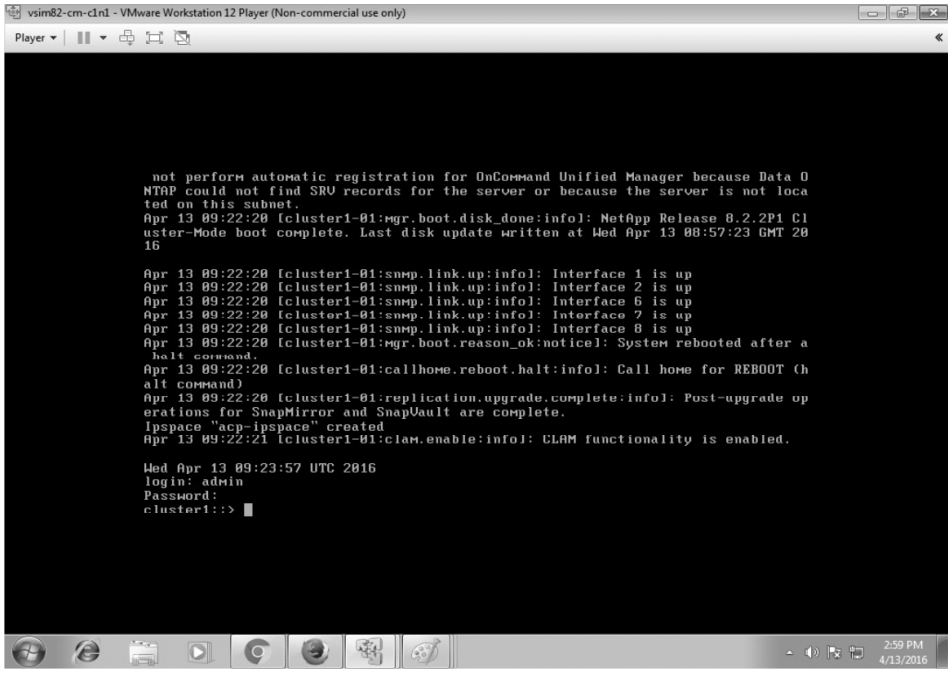

Figure 3: NetApp Simulator

2. Sequence Steps

2.1 Initially, CoprHD needs to find the NetApp Cluster information which includes basic system information, storage pools configured, virtual servers defined on the cluster and the network interfaces (storage ports).

2.2 After the successful discovery, the CoprHD system administrator would add these storage system to virtual array.

2.3 Once the CoprHD administrator groups the physical resources to a virtual array, he would create a virtual pool for the storage services-such as creation of snapshot, backup, remote replication, data reduction, encryption, etc.

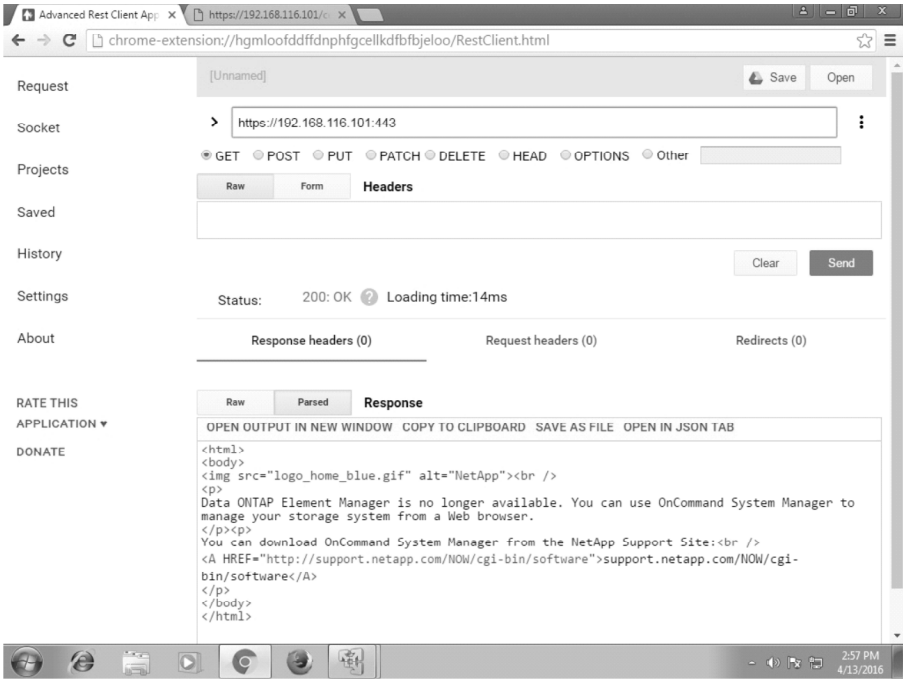

Figure 4: REST Client

2.4 Once the CoprHD system administrator defines the virtual pools, the tenant admin who creates the storage resources for users would use the virtual pool for storage services. The whole provisioning can be considered as three tasks-

2.4.1. Creation of file system- It is an asynchronous task with a POST request on /file/filesystems REST API. It takes the input parameters (name of file system, virtual array, virtual pool and size of the file system) as payload. The CoprHD apisvc validates the input parameters. Upon finding the parameters are valid, the apisvc creates a task for it and sends the request to controllersvc to process the actual file system creation. The controller service sends the request to respective filer drivers based on matched storage pool identified in placement matchers.

2.4.2. Provisioning of file system- Once the CoprHD controller decides that the file system needs to be created from NetApp, it sends the request to NetApp API. Earlier, CoprHD used the NetApp driver from iwave project. The iwave project in turn used the manageontap API to service provisioning requests. Presently, all discovery and provisioning tasks are done through NetApp REST API instead of using iwave project, i.e. each and every function in NetApp API is implemented using NetApp REST API.

2.4.3. Exporting file system- Exporting the file system is a process of masking the file system to client or user. This is an asynchronous task which uses POST request on /file/filesystems/{id} with export parameters as input. Similar to file system creation, this task also follows similar control flow.

V. CONCLUSION

The continuing growth in traditional enterprise application workloads coupled with the explosive growth in Web, mobile and cloud applications demands a simple, automated way to align data centre and storage resources to the numerous ways in which content is stored, protected and accessed.

The benefits of CoprHD are offered across multiple constituencies-

- Eliminate lock-in with multi-vendor support
- provides storage-as-a-service

- reduces cost
- centralized and automated management of storage
- promotes community driven development

New native Southbound API will be enabled for CoprHD third party driver developers. No changes to the existing device access interfaces in the controller will have to be done. The API will expose the same capabilities as we currently have in the native CoprHD device drivers.

ACKNOWLEDGMENT

We are indebted to Mr Shray Madan and Mr Mallari, EMC Academic Alliance for their support and motivation in successful implementation of this interface for CoprHD.

We are also thankful to Mr.Nimai Sood and Mr. Lakhinana Vasudevu for their continuous guidance throughout the completion of the project.

We would also like to thank Prof JSP Rai and Board of Management at Bharat Institute of Technology for funding our research without which this work would never have been possible.

REFERENCES

[1] A white paper on Copr HD Architecture, *https://CoprHD.atlassian.net/wiki/display/COP/A+Short+Guide+to+the+CoprHD+ Architecture ,Date of Access - 28 - July - 2016*.

[2] Mell, Peter and Timothy Grance. The NIST Definition of Cloud Computing (Draft). NIST. January 2011. *http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf*

[3] A white paper on Software Defined Storage, *http://www.infoworld.com/article/2610828infrastructure-storage/big-data-needs-software-defined-storage.html, Date of Access - 28 - July - 2016*.

[4] Cisco, 2013, Software-Defined Networking: Why We Like It and How We Are Building On It , *http://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/cis13090_sdn_sled_white_paper.pdf , Date of Access - 28 - July - 2016*.

[5] A white paper on advantages of Software Defined Networking, Glenn Brown, Ingram Micro, *http://www.ingrammicroadvisor.com/data-center/7-advantages-of-software-defined-networking, Date of Access - 28 - July - 2016*.

[6] Review on Virtualization for Cloud Computing D.Kiran Kumar, T.P.Sarachandrica,B.Rajasekhar,P.Jayasankar, ISSN (Online) : 2278-1021 ISSN (Print) : 2319-5940 International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 8, August 2014 *http://www.ijarcce.com/upload/2014/august/IJARCCE3A%20a%20sunil%20Review%20on%20Virtualization%20for%20Cloud%20Computing.pdf*

[7] A Review Paper on Virtualization and Security in Cloud Computing, Yogesh Bhardwaj, Dr. Manju Kaushik:Volume 4, Issue 3, March 2014 ISSN: 2277 128X, International Journal of Advanced Research in, Computer Science and Software Engineering, *http://www.ijarcsse.com/docs/papers/Volume_4/3_March2014/V4I3-0310.pdf*

[8] A white paper on ViPR Software defined storage from EMC[2,] *http://www.emc.com/collateral/white-papers/h11749-transform-data-center-with-vipr-software-defined-storage-wp.pdf, Part Number H11749.5, May 2015, Date of Access - 28 - July - 2016*.

[9] A data sheet on ViPR SDS, *http://www.emc.com/collateral/data-sheet/h11750-emc-vipr-software-defined-storage-ds.pdf, Date of Access - 28 - July - 2016*.

[10] A white paper on Integration of Storage Device Drivers, *https://CoprHD.atlassian.net/wiki/display/COP/Integration+of+Storage+Device+Drivers, Date of Access - 28 - July - 2016*.